

ASTAS-DLL

Bedienungsanleitung

A.S.T. Angewandte System-Technik GmbH
Marschnerstraße 26 01307 Dresden
Telefon (03 51) 44 55 30 Telefax (03 51) 44 55 555
www.ast.de astmr@ast.de

Inhaltsverzeichnis

INHALTSVERZEICHNIS	1
1 ALLGEMEINES.....	2
2 DATENSTRUKTUREN	3
3 FUNKTIONEN	5
4 RÜCKGABEWERTE DER FUNKTIONEN.....	11
5 STATUS-BYTE	12
6 MASSEINHEITEN	13
7 GERÄTETYPEN.....	13
8 SONSTIGE DEFINES	13

1 ALLGEMEINES

Es können max. 16 Geräte (AE703 / BD341/2 / BA661/2 / FFB201 / FFB204) an der USB-Schnittstelle durch die ASTAS-DLL verwaltet bzw. angesprochen werden.

Die Geräte werden in einem Strukturarray (siehe Kap. 2, *t_DeviceInfo*) verwaltet. Der Listenindex *list_idx* geht von 0...15. Die einzelnen Geräte werden über diesen Listenindex (siehe Kap.3, *list_idx*) angesprochen.

Aktuelle Messerte werden über eine der beiden Strukturen (siehe Kap. 2, *t_DeviceData* und *t_DeviceDataFFB*) abgefragt. Darüber hinaus gibt es Funktionen zur Steuerung der Geräte.

Die Definitionen der Datenstrukturen, Funktionsprototypen und der einzelnen Konstanten ist in folgenden Dateien zu finden.

astas_dll.h -> allg. C-Header-Datei

astas_DLLImport.h -> spez. Header-Datei für Microsoft Visual C++
(DLL-Import Header-Datei)

Hinweis 1:

Für den Einsatz der ASTAS-DLL ist es von Vorteil die Software ASTAS zu installieren, um Einstellungen an den Geräten vornehmen zu können.

<http://www.ast.de/files/downloads/mess-und-regeltechnik/controller/ASTAS.zip>

Die jeweils aktuelle Version der ASTAS-DLL ist ebenfalls im Downloadbereich zu finden.

http://www.ast.de/files/downloads/mess-und-regeltechnik/controller/ASTAS_DLL.zip

Hinweis 2:

Ab ASTAS-DLL V0.12.x hat sich die Reihenfolge jeweils in den Strukturen der Messwerte verändert! (siehe Kapitel 2 - Datenstrukturen)

2 DATENSTRUKTUREN

Struktur der Gerätedaten

```
typedef struct
{
    DWORD usb_hid_idx;           // USB-Geraete-Index -> interne Verwendung!
    int open;                    // Zustand -> 1=Open / 0=Close
    char vid_pid[256];           // VID/PID-String
    char dev_info[256];          // GeräteInfo-String
    char sn_info[256];           // S/N-String
    int hw_info;                 // HW-Info (5xxx)
    unsigned char hw_var;        // HW-Variante
    int fw_vers;                 // FW-Version
} t_DeviceInfo;
```

Struktur der Messwerte (AE703 / BD341/2 / BA661/2)

```
typedef struct
{
    float brutto;                // Brutto / Messwert
    float netto;                 // Netto
    float tara;                  // Tara
    float min;                   // Min (bezogen auf aktuelle An-Zeit)
    float max;                   // Max (bezogen auf aktuelle An-Zeit)
    unsigned char me;            // Maßeinheit
    unsigned char dec;           // Anzahl Nachkommastellen
    unsigned char status;        // Status-Byte
} t_DeviceData;
```

Struktur der Messwerte (FFB201 / FFB204-A...D mit idx (siehe Hinweis 2))

```
typedef struct
{
    float mw[4];                 // Messwert / Anzeigewert
    unsigned char me[4];         // Maßeinheit
    unsigned char dec[4];        // Anzahl Nachkommastellen
    unsigned char status[4];     // Status-Byte
} t_DeviceDataFFB;
```

Die Deklaration der Datenstruktur/des Datenarrays für max. 16, durch die ASTAS-DLL verwaltete, Geräte kann folgendermaßen aussehen.

```
t_DeviceInfo    myDeviceInfo[16];           // list_idx = 0...15
```

Diese muss vom aufrufenden Programm intern angelegt werden. Der Zeiger auf diese Datenstruktur bzw. das Datenarray wird dann an die Search-Funktion übergeben.

Hinweis 1:

usb_hid_idx entspricht nicht list_idx und wird nur für interne Zwecke verwendet!

Hinweis 2:

Die Datenstruktur `t_DeviceDataFFB` enthält die Messwerte der FFB201 bzw. FFB204.

- `idx=0:` FFB201 bzw. FFB204-A
- `idx=1:` FFB204-B
- `idx=2:` FFB204-C
- `idx=3:` FFB204-D

Hinweis 3:

Die verwendeten Datentypen entsprechen bei einer 32bit-DLL:

DWORD	32bit	INT32	signed int
int	32bit	INT32	signed int
char	8bit	INT8	signed char
unsigned char	8bit	UINT8	unsigned char

3 FUNKTIONEN

```
extern "C" DLL_API int AstasDllInit(void);
extern "C" DLL_API int AstasDllReInit(void);
extern "C" DLL_API int AstasDllSearch(t_DeviceInfo *p_dev_info);
extern "C" DLL_API int AstasDllOpen(int list_idx);
extern "C" DLL_API int AstasDllClose(int list_idx);
extern "C" DLL_API int AstasDllReadData(int list_idx,
                                       t_DeviceData *dev_data);
extern "C" DLL_API int AstasDllReadDataFFB(int list_idx,
                                       t_DeviceData_FFB *dev_data_ffb);
extern "C" DLL_API int AstasDllNull(int list_idx);
extern "C" DLL_API int AstasDllTara(int list_idx);
extern "C" DLL_API int AstasDllGrossNet(int list_idx);
extern "C" DLL_API int AstasDllMax(int list_idx);
extern "C" DLL_API int AstasDllChangeUnit(int list_idx);
extern "C" DLL_API int AstasDllChangeRange(int list_idx);
extern "C" DLL_API int AstasDllAckChangeStatusBits(int list_idx);
extern "C" DLL_API int AstasDllCalibNull(int list_idx);
extern "C" DLL_API int AstasDllCalibEnd(int list_idx, float cal_val);
extern "C" DLL_API int AstasDllCalibSave(int list_idx);
extern "C" DLL_API int AstasDllCalibDelete(int list_idx);
extern "C" DLL_API int AstasDllSetMinMax(int list_idx, int mode);
extern "C" DLL_API int AstasDllReset(int list_idx);
extern "C" DLL_API int AstasDllDLLVersion(void);
```

Hinweis 1: Abhängig von der Firmwareversion des angeschlossenen Gerätes und des gewählten Gerätes können Funktionen nicht implementiert sein.

Hinweis2: Ab ASTAS-DLL V0.10.0 sind die Funktionen zum Lesen der Daten
- ReadData und ReadDataFFB - getrennt für AE703 / BD341/2 / BA661/2 bzw. FFB201 / FFB204 zu verwenden.

Hinweis3: Ab ASTAS-DLL V0.11.0 haben alle Funktionen einen vorgestellten Präfix AstasDll... erhalten!

int AstasDllInit(void)

Initialisierung der ASTAS-DLL. Dieser Schritt ist notwendig für eine korrekte Funktion der ASTAS-DLL.

Rückgabe: NO_ERR / ERR

Hinweis: Diese Funktion muss zwingend als erste Funktion zur Initialisierung der DLL aufgerufen werden.

int AstasDllReInit(void)

Re-Initialisierung der ASTAS-DLL.

Rückgabe: NO_ERR / ERR

int AstasDllSearch(t_DeviceInfo *p_dev_info)

Mit dieser Funktion werden alle angeschlossenen Geräte AE703 / BD341/2 / BA661/2 / FFB201 / FFB204 (max. 16) gesucht und die Informationen in der DeviceInfo-Struktur abgelegt. Diese Struktur muss vom aufrufenden Program als Zeiger/Pointer übergeben werden.

Rückgabe: Anzahl der gefundenen Geräte

Hinweis: Ab ASTAS-DLL V0.9.9 wird zusätzlich an den GeräteInfo-String (s.o.) der Gerätenamen aus „Setup -> Gerät-> Gerätebezeichnung“ (siehe ASTAS) nach einem Trennzeichen (Semikolon / „;“ bzw. 0x3B mit Leerzeichen / „ „ bzw. 0x20) angehängt.

Geräte-Info (ab ASTAS-DLL V0.9.9):

AE703: "AST_AE703"
BD341/2: "AST_BD342"
BA661/2: "AST_BA662"
FFB201: "AST_FFB201"
FFB204: "AST_FFB204"

Beispiel: „AST_AE703; AE703_1 „

B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	T0	T1	S0	S1	S2	S3	S4	S5	S6	S7
	A	S	T	_	A	E	7	0	3	;		A	E	7	0	3	_	1	

B0..9: 10 Zeichen/Bytes interner Gerätebezeichner
(linksbündig mit Leerzeichen aufgefüllt)
T0: Trennzeichen(1) (Semikolon/0x3B)
T1: Trennzeichen(2) (Leerzeichen/0x20)
S0..7: 8 Zeichen/Bytes des Setup-Gerätebezeichners
(rechtsbündig mit Leerzeichen aufgefüllt)

`int AstasDllOpen(int list_idx)`

Öffnen des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

`int AstasDllClose(int list_idx)`

Schliessen des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

**`int AstasDllReadData(int list_idx,
t_DeviceData *p_dev_data)`**

Lesen der aktuellen Messserte(s) des gewählten Gerätes. Es muss die Struktur t_DeviceData als Zeiger/Pointer übergeben werden.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

**`int AstasDllReadDataFFB(int list_idx,
t_DeviceDataFFB *p_dev_data_ffb)`**

Lesen der aktuellen Messserte(s) des gewählten Gerätes. Es muss die Struktur t_DeviceDataFFB als Zeiger/Pointer übergeben werden.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: FFB201 / FFB204

Hinweis:

Als Rückgabe erfolgt der aktuelle Messwert/Anzeigewert des mobilen Gerätes.

`int AstasDllNull(int list_idx)`

Nullung des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

Hinweis:

Eine nicht erfolgte Nullung wird nicht als fehlerhaft zurückgegeben.

Eine Nullung kann über die Rückgabe des Status-Bytes bei Lesen des aktuellen Messwertes kontrolliert werden.

int AstasDllTara(int list_idx)

Tarierung des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: FFB201 / FFB204

int AstasDllGrossNet(int list_idx)

Umschaltung zwischen Anzeige Brutto/Netto des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / FFB201 / FFB204

int AstasDllMax(int list_idx)

Umschaltung zwischen Anzeige Brutto/Netto des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: FFB201 / FFB204

int AstasDllChangeUnit(int list_idx)

Umschaltung der Maßeinheit des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

int AstasDllChangeRange(int list_idx)

Umschaltung des Messbereichs des gewählten Gerätes.

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

int AstasDllAckChangeStatusBits(int list_idx)

Bestätigung der Statusbits ChangeUnit/ChangeRange. Gerät setzt darauf hin beide Statusbits zurück.

Rückgabe: NO_ERR / Fehlercode

Geräte: AE703 / BD341/2

int AstasDllCalibNull(int list_idx)

Lastkalibrierung Nullpunkt (fest 0.0 = 0%).

Rückgabe: NO_ERR / Fehlercode

Geräte: AE703 / BD341/2 / BA661/2

int AstasDllCalibEnd(int list_idx, float cal_val)

Lastkalibrierung Endpunkt (normierter variabler Wert vom Nennkennwert (100%)).

cal_val: 1.0 = 100%
0.5 = 50%
0.0 = 0%

Rückgabe: NO_ERR / Fehlercode

Geräte: AE703 / BD341/2 / BA661/2

int AstasDllCalibSave(int list_idx)

Lastkalibrierung abschließen und im Gerät speichern.

Rückgabe: NO_ERR / Fehlercode

Geräte: AE703 / BD341/2 / BA661/2

int AstasDllCalibDelete(int list_idx)

Lastkalibrierung im Gerät löschen. Notwendig vor jedem Kalibriervorgang!

Rückgabe: NO_ERR / Fehlercode

Geräte: AE703 / BD341/2 / BA661/2

int AstasDllSetMinMax(int list_idx, int mode)

Setzen der min/Max-Werte auf aktuelle Meßwerte.

mode: 1 – MODE_SETMIN
2 – MODE_SETMAX

Rückgabe: NO_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

int AstasDllReset(int list_idx)

Reset des gewählten Gerätes. Danach müssen die Geräte neu gesucht und geöffnet werden.

Rückgabe: NO_ERR

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

int AstasDllDLLVersion(void)

Abfrage der ASTAS-DLL-Version. Rückgabe ist ein int-Wert, der folgende Informationen enthält.

Byte3: Hauptversion

Byte2: Unterversion

Byte1: Revisionsversion

Byte0: Reserve (int. Zähler)

Beispiel:

Hex: 0x000C000yy – Vers. 0.12.0.yy

4 RÜCKGABEWERTE DER FUNKTIONEN

NO_ERR	0	kein Fehler
ERR	1	allg. Fehler (z.B. Gerät nicht ansprechbar)
ERR_DEVICE_NOT_OPEN	10	Gerät nicht geöffnet
ERR_FUNC_NOT_IMPL	20	Funktion nicht implementiert (abh. von der FW-Version des Gerätes)
ERR_DEVICE_READ	30	Lesefehler von Gerät (z.B. Gerät von USB getrennt)
ERR_FFB_RF	40	Funk-Fehler (FFB20x)
ERR_ONE_DEVICE_OPEN	50	Fehler bei Öffnen Gerät (wenn ein Gerät geöffnet ist)
ERR_DEVICE_READ_FALSE	60	Lesefehler (FFB20x <-> andere Geräte)
ERR_DEVICE_CALIB_MVV	100	bei Maßeinheit mV/V (UNIT_mV_V) keine Kalibrierung möglich
ERR_DEVICE_CALIB_READ_VAL	101	erst Messwert lesen, dann werden Kalibrier-Funktionen ausgeführt
ERR_DLL_REINIT	250	Fehler bei ReInit der ASTAS-DLL
ERR_DLL_MEM_INIT	251	Speicherfehler beim init. der ASTAS-DLL
ERR_DLL_CURR_INIT	252	DLL ist schon initialisiert
ERR_DEVICE_FALSE_IDX	253	falscher Wert für <i>list_idx</i>
ERR_DEVICE_NOT_EXIST	254	gewähltes Gerät nicht vorhanden
ERR_DLL_NOT_INIT	255	ASTAS-DLL wurde nicht initialisiert

5 STATUS-BYTE

Bei Abfrage der Messwerte wird das Status-Byte zurückgegeben.
Die einzelnen Bitpositionen sind folgend erklärt.

AE703 / BD341/2 / BA661/2

-Reserviert-	0x01	0b00000001	// Reserviert
STATUS_NULL	0x02	0b00000010	// Status -> Null-Anzeige // (Netto-Wert)
STATUS_TARA	0x04	0b00000100	// Status -> Tara/Null
STATUS_OVER	0x08	0b00001000	// Status -> Ueberlast
STATUS_UNDER	0x10	0b00010000	// Status -> Unterlast
STATUS_CHG_UNIT	0x20	0b00100000	// Status -> Änd. Maßeinheit
STATUS_CHG_RANGE	0x40	0b01000000	// Status -> Änd. Messbereich
STATUS_ERR	0x80	0b10000000	// Status -> ADC-Error

FFB201 / FFB204

-Reserviert-	0x01	0b00000001	// Reserviert
STATUS_NULL	0x02	0b00000010	// Status -> Null-Anzeige // (Netto-Wert)
STATUS_TARA	0x04	0b00000100	// Status -> Tara/Null
STATUS_OVER	0x08	0b00001000	// Status -> Ueberlast
STATUS_UNDER	0x10	0b00010000	// Status -> Unterlast
STATUS_MAX	0x20	0b00100000	// Status -> Max.-Wert
STATUS_RF_IO	0x40	0b01000000	// Status -> RF i.O.
-Reserviert-	0x80	0b10000000	// Reserviert

6 MASSEINHEITEN

Die Bedeutung des Maßeinheiten-Byte ist im Folgenden erklärt

UNIT_USER	0	// User-Einheit	(Definierbar im Gerät bzw. per ASTAS-Software)
UNIT_N	1	// N	
UNIT_kN	2	// kN	
UNIT_g	3	// g	
UNIT_kg	4	// kg	
UNIT_t	5	// t	
UNIT_lbf	6	// lbf/Pfund	
UNIT_oz	7	// oz/Unze	
UNIT_mV_V	8	// mV/V	
UNIT_M300	9	// M300	
UNIT_M600	10	// M600	
UNIT_to	11	// to	(amerikanische Tonne)

7 GERÄTETYPEN

AE703	5100
BD341/2	5140
FFB201	5200
FFB204	5240
BA661	5310
BA662	5330

8 SONSTIGE DEFINES

MODE_SETMIN	1	// Setzen des Min-Wertes auf aktuellen Meßwert
MODE_SETMAX	2	// Setzen des Max-Wertes auf aktuellen Meßwert

Benötigt für Funktion SetMinMax().